Revised Project Proposal
Tony Faller (af3370)

Team Name: MassTrack
Project Name: MassTrack: The MBTA Green Line

Please be aware that this project proposal is an extension of a project that is concurrently being developed in another course: COMS E6998 Private Systems with Professor Roxana Geambasu. I've received her permission to extend this project in this context.

Link to Other Proposal: [Privacy Attacks - Public Data] MassTrack
(I believe this Google Doc can only be accessed with a Lionmail account.)

The Private Systems Project
Earlier this semester in Professor Geambasu's Private Systems course, I learned about a privacy study that recent students performed. They examined publicly-available CitiBike data which included fields such as startStation, stopStation, startTime, and stopTime. Those students framed their research in the following context:

- Given start station/time, can end station/time be determined?
- Is this information safe for public release?
    - Consider the case of stalking/harassment; if an adversary sees a person leave from a station and notes down the time, they may be able to determine exactly where that person is going

Those students found that a certain percentage of records were uniquely identifiable given start station/time. They also found that certain side information, such as frequent stations or co-riders, enabled tracking.

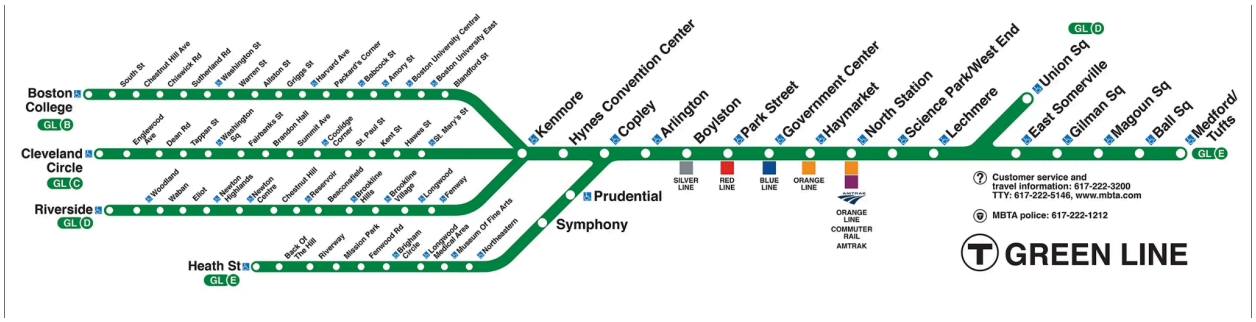
In Private Systems, I am extending this prior work by performing the same attack on BlueBike data (BlueBike is the same exact thing as CitiBike, including being owned by BikeShare/Lyft, but implemented in Boston) to see how widely this attack can apply. I am executing this project as a solo group, though I do have the assistance of the Private Systems teaching staff available.

~~As of this proposal's submission, no steps have been taken beyond proposing this project, finding appropriate datasets, and ensuring that the terms-of-service permit privacy research.~~

As of this revised project proposal's submission, I have worked towards a basic analysis of BlueBike data. Further detail is available via the link above, but I do not believe this will have any impactor nor overlap with this project.

This Project
In the context of this class, I would like to extend this work by performing a similar attack on subway data. Subway data is fundamentally different from bike data, as bike data is

point-to-point whereas subway data is point-to-[anywhere on the line, including transfers and direction change]. I anticipate that the attack would require additional side information, such as time spent in transit.
Due to time constraints, I plan to limit myself to Boston's Green Line. The subway would be represented as a graph, with one node per station, edge weights representing time between stations, and special "transfer nodes" representing stations at which passengers are likely to change directions or lines. Edge weights will be determined at runtime by matching user input to existing data. For simplicity, transfers between Boston's lines (i.e. Green to Red, Green to Orange, etc) will not be considered. I've written up a rough overview of the system in appendices at the bottom of this document, though this extends the proposal beyond the 1-2 page limit.



Boston's Green Line, part of the T subway system

With a provided input of start station/time and time-in-transit, the MassTrack system will output a list of possible end stations. To evaluate this system, I plan to use the following metrics:

Efficiency: How long does it take to determine the possible stop locations?

- Metric: Execution Time
- Visualization: Plot of execution time vs time-in-transit
- Compare against a naive implementation which assumes the same graph but with constant edge weights

Accuracy: Is the true end station within the list of possible end stations?

- ~~Preserve privacy via constructing synthetic data; include relevant events like transfers, different start times, and varying true transit times~~
- Synthetic data is not required; see appendix for further detail.
- Metric: Boolean; true if for given input, correct end station is within output
    - Could construct a handful (10+) of synthetic rows and report success rate
- Metric: 1 / (# output end stations) would represent accuracy as well
    - Could plot this accuracy metric against input time-in-transit
- I intuitively anticipate that accuracy decreases with increasing time-in-transit.

A low execution time coupled with a high degree of accuracy would indicate that public subway data may inherently have associated privacy risks, despite the non-existence of personal

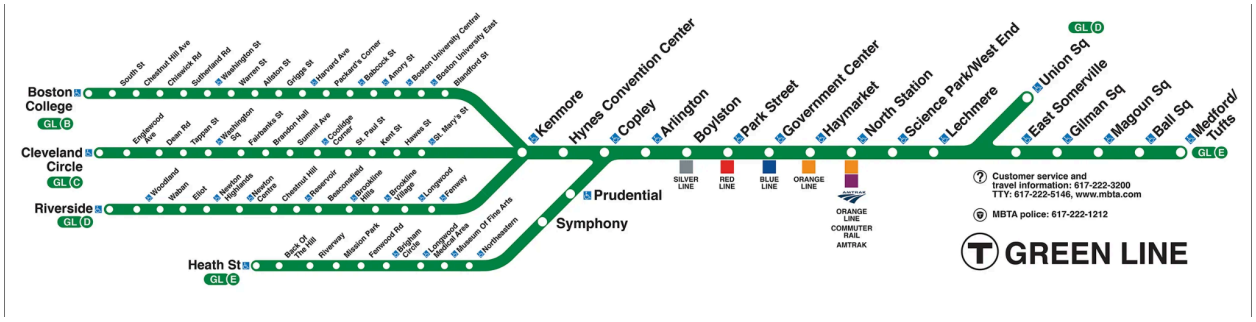
identifying information. I was not able to find prior research which explicitly attacks public subway data for the purpose of tracking, which would indicate that this project would be novel.

Appendix: The Green Line

Boston's public transit is managed by the Massachusetts Bay Transit Authority (MBTA). The train system is known as the T, and it is broken down into multiple color-named lines. The MBTA publishes origin/destination data for the T; for example, here is a link to 2022 data: https://mbta-massdot.opendata.arcgis.com/datasets/e48c8cd8cf154d27b23d31777a8f39e9/about

The Green Line is one of these lines, and it is further decomposed into multiple letter-named lines. This is visualized below. Despite its seemingly linear appearance, not all lines can get to all stations; for example, a B line train only travels to and from the Boston College stop and the Government Center stop. The lines are enumerated:

B: Boston College ↔ Government Center
C: Cleveland Circle ↔ Government Center
D: Riverside ↔ Union Square
E: Heath Street ↔ Medford/Tufts



Boston's Green Line, part of the T subway system

Some observations:

- West of the Kenmore stop: the B, C, and D lines are isolated.
- East of the Kenmore stop: the B, C, and D lines all run on the same line and stop at the same stations.
- Both of the above are true for the E line, except at the Copley stop instead.
- East of Government Center & West of Lechmere: the D and E lines run on the same line.
- East of Lechmere: the D and E lines are isolated.

Abstracting the Green Line

A graph can be constructed to represent the Green Line. Each node is a station, and edges represent connections between stations. Edge weights will represent the transit time between stations, per public data.

So-called “Transfer Nodes” will represent stations at which a rider is likely to transfer to a different train, as opposed to staying on the train they are already on. This would occur when a rider needs to change directions or lines, i.e. someone travelling from BU East on the B line to Waban on the D line would likely transfer at Kenmore.

Transfer Nodes

- Kenmore
- Copley
- Government Center
- Lechmere

Note that there are transfers available from the Green Line to the other color T lines. For simplicity, it is assumed that the rider stays within and does not transfer off of the Green Line.

Appendix: Rough Sketch of Algorithm

User Input: Start Station, Start Time, Time-in-transit

Constants

- C1: A constant time interval used to determine the closest record to user’s start time

Algorithm Details

1. Initialize time budget T, start node A, and datapoint D
    a. T ← a time interval equal to the time-in-transit parameter; keeps track of cumulative transit time
    b. A ← the graph node which corresponds to the start station parameter
    c. D ← the closest record in the dataset which matches the start station and is within C1 of start time

2. If A is a transfer node:
    a. Identify all possible transfer lines
    b. Execute the following steps in parallel for each transfer line, with each keeping track of an independent time budget

3. Determine the next datapoint D’ which:
    a. Is on the same train line; and
    b. Has an arrival time greater than D’s departure time

4. Determine the node A’ which corresponds to the station of datapoint D’

5. Set $T = T - (D'_{arrive} - D_{depart}) - (D_{depart} - D_{arrive})$
    a. D_{arrive} ← Timestamp of then the train arrived at station A
    b. D_{depart} ← Timestamp of then the train departed station A
    c. D'_{arrive} ← Timestamp of when the train arrived at station A’
    d. $D'_{arrive} - D_{depart}$ ← Interval representing time spent travelling between stations A and A’
    e. $D_{depart} - D_{arrive}$ ← Interval representing time the train spent in station A

6. Set D = D’ and A = A’
7. Repeat steps 2-5 until budget is exhausted
8. Output A

Appendix: Why Synthetic Data is Not Required

In the original project proposal, I mentioned that I would use synthetic data to preserve privacy while assessing the accuracy of the MassTrack system. I now no longer believe that this is necessary. I had concerns that this analysis would constitute a reidentification attack; however, there is a subtle nuance which can be summed as follows: “knowing that person A left Station C at time T is different from knowing that train B left Station C at time T.”

As-proposed, the MassTrack system *could* be used to perform a reidentification attack. For example, say that an adversary knows that a person regularly leaves a particular station at a certain time and spends a certain amount of time in transit. The adversary could enter these inputs into the MassTrack system and determine possible end stations for where that person went – i.e. *reidentify* this person in the anonymous non-user-affiliated data. MassTrack is NOT the adversary in this context.

The MassTrack system itself amounts to a timing analysis of the train system. This project highlights that such a timing analysis can enable reidentification (given side information about a target) and demonstrates that this data is vulnerable to attack.